System Identification by Genetic Algorithm

Vu Duong
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109
vduong@brain.jpl.nasa.gov

Allen R. Stubberud
ECE Department
University of California, Irvine
Irvine, CA 92697
arstubbe@uci.edu

Abstract
This paper presents a method for identifying systems through their input-output behavior and the Genetic Algorithm. The advantages of this technique are, first, it is not dependent on the deterministic or stochastic nature of the systems and, second, the globally optimized models for the original systems can be identified without the need of a differentiable measure function or linearly separable parameters. The results are compared to similar results from Least Squares identification methods.

1. Introduction

The problem of determining a mathematical model for an unknown system by observing its input-output data pairs is generally referred to as system identification [1]. System identification is performed by adjusting parameters within a given model until its output, for a particular input, coincides as well as possible with the measured output of the system being identified for the same input. After a system has been identified, its output can then be predicted for a given input to the system. This, of course, is usually the primary goal of the system identification problem.

In general, system identification involves two steps: structure realization and parameter identification. In the first step, a priori knowledge is used to determine a class of models to which the target system may belong. For example, the target system might be assumed to be causal, time-invariant, etc. If there is no a priori knowledge available, then the structure realization might be done by a trial and error method [2]. The main focus in system identification is on the parameter identification process.
Consider a mapping,

$$G: U \rightarrow V$$

where G is a system that maps a set of inputs U to a set of outputs V and for which an approximation is desired.

Let $\hat{G}$ be an approximation for G, where our interest is only in external approximation, that is, where the approximation objectives are focused on the behavior of the approximation error

$$e(t) = (Gu)(t) - (\hat{G}u)(t) \tag{1}$$

Such approximation objectives are called external approximation criteria [3].

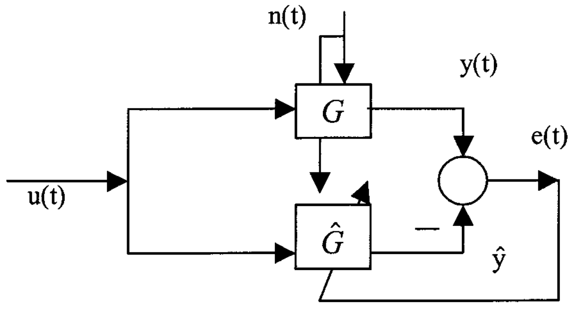


Figure 1: a block diagram for system identification, G is a target system, $\hat{G}$ is estimated system.

The approximation criterion allows us to determine how good the estimate of the system is.

For example, one important external approximation criterion is the uniform approximation criterion which states that the absolute approximation error |e(t)| must be uniformly less than a prescribed ε [3].

Several techniques have been used to minimize the approximation error, e(t), and most are based on statistical or gradient descent methods. Statistical methods require knowledge of the statistical properties of the systems as well as of the input and output signals. Such information is sometimes not available. Gradient descent methods are local search techniques, which require a smooth search space. Even when the search space is smooth, these techniques are often unable to locate the global optimum because of the presence of multimodal error surfaces and/or because of the problem of dimensionality when identifying high order systems. In this paper, we approach the problem of minimizing e(t) by using the Genetic Algorithm. Advantages of the Genetic Algorithm are that it generally requires less knowledge of system properties and statistical properties of the input-output signals, is a global search technique, and avoids the problem of dimensionality.

2. Genetic Algorithm

The Genetic Algorithm (GA) is a two-step population-based process of random variation and selection [4]. In GA, a string of a certain data type, where the data type can be numeric, binary or a user-defined type represents a solution. The string structure is called a chromosome. For example, in binary coding, each parameter is coded using a binary string of l bits whereby a parameter can take values lying in the interval [0 2^l-1]. A linear mapping procedure is used to decode any unsigned integer from [0 2^l-1] to a specific interval [Low, High]. For multiparameter optimization, the coded parameter values are concatenated to form a large string, which then forms one member (chromosome) of a population as illustrated below:

α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8
11010111	01110010	1111101	00011100	11111001	00111101	10010001	11001000

A coded string of eight parameters

The concept of a genetic algorithm is that a collection of potential solutions to a problem is created by taking random numbers from a chosen distribution and then using crossover and mutation operators to generate new solutions. A crossover operator is applied to a pair of solution strings (parents) by exchanging a part of one string with another part of the other string to create two new solution strings (children). A mutation operator operates on only one string (a parent) by negating one part of or the whole string, thus creating a new string (a child). These two operators are illustrated below:

parent	10111111111111100000|**100001000**|10001	
parent	010000|***001011111***|1011111011100001111	10111111111111100000|**1000010010001**
child1	10111111111111100000|***001011111***|10001	10111111111111100000|**0111101101110**
child2	010000|**100001000**|1011111011100001111	

New children created by crossover operator A new child created by mutation operator

A selection criterion is now imposed to determine which solution should be kept and which should be discarded. The selection sub-process goes through several steps as follows: first, each solution in the collection is evaluated and scored based on a given evaluation criterion. Second, the potential solutions are ranked based on fitness scores. From these rankings, the selection process chooses solutions to be advanced to the next generation. To assign a fitness score to each string structure, the string is decoded into its constituent parameters. The error signal obtained by using these parameters is used in calculating the fitness measure for the string. There are several ways to define fitness score using the error signal. However, the mean square value of the error signal over a rectangular window is often used as the fitness score for the string [5]. This two-step process is then repeated until a stopping criterion is satisfied. The stopping criteria might be a maximum number of generations that the evolving process will go through, or it might be that the score of a potential solution must lie within a certain boundary. Figure (2) is a flow chart of a general GA.

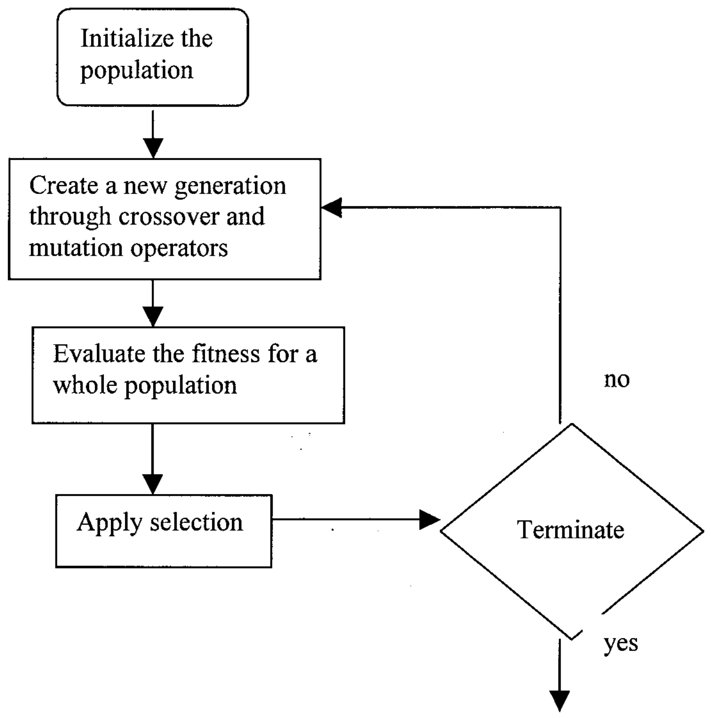

Initialize the population
Create a new generation through crossover and mutation operators
Evaluate the fitness for a whole population
Apply selection
Terminate
no
yes

Figure 2: a general flow chart of genetic algorithm.

3. Linear system identification

In this section, the GA is applied to the identification of a linear system. Here, we assume that the structure of a system is in the form of an auto regressive (AR) and moving average (MA), and that the number of lag terms to be included in the AR is known.

Consider a system as follows:

$$\begin{aligned} y[t+T_s] &= a_0 y[t] + a_1 y[t-T_s] + \cdots + a_M y[t-MT_s] \\ & b_0 u[t] + b_1 u[t-T_s] + \cdots + b_N u[t-NT_s] \\ & c_0 n[T] + c_1 n[t-T_s] + \cdots + c_L n[t-LT_s] \end{aligned} \qquad (2)$$

where $y[t]$, $u[t]$ are the output and input of the system respectively; $n[t]$ is a white Gaussian noise with zero mean and unit variance and there is no correlation between $n[t]$ and the observed output. The unknown system coefficients are $[a_0\ a_1 \ldots a_M\ b_0\ b_1\ b_N\ c_0\ c_1 \ldots\ c_L]$ and T_s is the sampling time.
In shorthand notation, the model (1) can be written as:

$$\mathrm{A}(q^{-1})y[t] = B(q^{-1})u[t] + \mathrm{C}(q^{-1})n[t] \qquad (3)$$

where $\mathrm{A}(\mathrm{q}^{-1})$, $\mathrm{B}(\mathrm{q}^{-1})$, $\mathrm{C}(\mathrm{q}^{-1})$ are polynomials of the unit delay operator q^{-1}.

Assume that the approximation, $\hat{y}$ (t), the observed values of $y[t]$ and $u[t]$, and the statistical parameters of $n[t]$ are all known, then a performance measure **J** for the system to be identified is defined as follows:

$$
\begin{aligned}
\mathrm{J} = \int_{t_1}^{t_2} (\hat{y}[t+T_s] - a_0\hat{y}[t] - a_1\hat{y}[t-T_s] - \cdots - a_M\hat{y}[t-MT_s] \\
- b_0u[t] - b_1u[t-T_s] - \cdots - b_Nu[t-NT_s] \\
- c_0n[T] - c_1n[t-T_s] - \cdots - c_Ln[t-LT_s]\,)^2\,\mathrm{dt}
\end{aligned} \qquad (4)
$$

The objective is to choose $[a_0\, a_1 \ldots a_M\, b_0\, b_1 \ldots b_N\, c_0\, c_1 \ldots c_L]$ so that J is minimized
The ideal value for J would, of course, be zero

It is known that pole and zero based identification may be more robust when zeros are included, because modeling errors are more sensitive to changes in system poles than zeros. Moreover, determining the searching ranges for the pole and zeros may be difficult if the system is unstable or non-minimum phase. Therefore, identification is often carried out on the coefficients of the transfer function, instead of pole and zeros positions [6].

A simulation of the identification of a linear system was carried on a linear discrete-time system. The observed outputs of the system were generated by the difference equation (5).

$$
\begin{aligned}
\hat{y}[k] = 1.5\hat{y}[k-1] - 0.7y[k-2] + \\
u[k-5] + 0.5u[k-6] + \\
n[k] - n[k-1] + 0.2n[k-2]
\end{aligned} \qquad (5)
$$

where the input signal u[k] = $[1.1]^{-k}$ and n[k] was a Gaussian white noise with zero mean and unit variance. The model system, whose parameters were identified, was in the form

$$
\begin{aligned}
y[k] = a_1y[k-1] + a_2y[k-2] + \\
b_1u[k-5] + b_2u[k-6] + \\
c_1n[k] + c_2n[k-1] + c_3n[k-2]
\end{aligned} \qquad (6)
$$

The performance measure for the model system was defined as in (4):

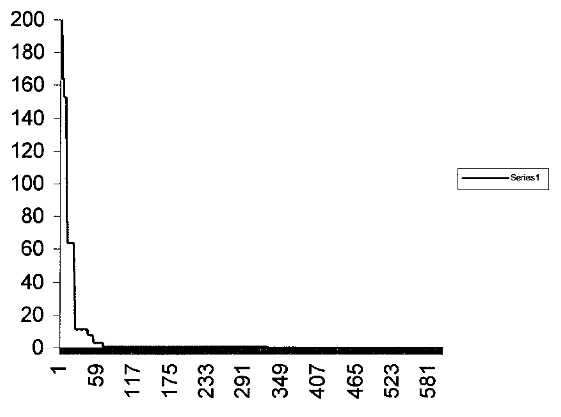
$$
\begin{aligned}
\mathrm{J} = \sum_{k=1}^{W} (\hat{y}[k] - a_1y[k-1] - a_2y[k-2] \\
- b_1u[k-5] - b_2u[k-6] \\
- c_1n[k] - c_2n[k-1] - c_3n[k-2])^2
\end{aligned} \qquad (7)
$$

where W =200 is the window size for the signals.

The objective now was to optimally find $[a_1\, a_2\, b_1\, b_2\, c_1\, c_2\, c_3]$ in the difference equation (6) such that J was minimized. An initial population of 500 models was generated randomly. Each model consisted of seven parameters uniformly distributed over the range of [-2, 2]. Each offspring was created uniformly by applying the crossover operator with probability of 0.7 and the mutation operator with the probability of 0.4 on the parents. The performance or fitness score for each model was defined in terms of the mean square error between the output of the target system and the output of the model as in equation (7). The selection scheme was applied as followed: 20% of the highest fitness scores of the current population survived to the next generation and the remaining 80% of the current population was replaced by newly created offspring. This process was iterated for 600 generations.

To show that the method is robust to a variety of inputs, the simulation was carried out with different inputs. Figure 3 indicates the error performance of the best model in each generation through the entire evolutionary-optimization process. Figure 3a) associates with input u[k] = $(1.1)^{-k}$ and figure 3b) associates with input u[k] = tanh(k). There was a rapid improvement of the best fitness score from one generation to the next generation at the beginning of the processes. In both cases, the processes converged to zero rapidly. However, with different inputs, the parameters obtained in the two cases are slightly different. The final best-evolved model after 600 generations with u(k) = $(1.1)^{-k}$ was [1.484 –0.6867 0.1606 1.2856 .971 -.7899 .0193], and the associated error was 6.4167e-2. The best evolved model with u(k) = tanh(k) was [1.467 0.6577 1.5904 –0.0813 0.8535 -.6479 –0.746], and the associated error was 8.827e-2.

The sequence *n*(*t*) was chosen as a Gaussian white noise in order for our model to be a good fit to a variety of input data sets. However, its statistical properties were not used in the optimization process.



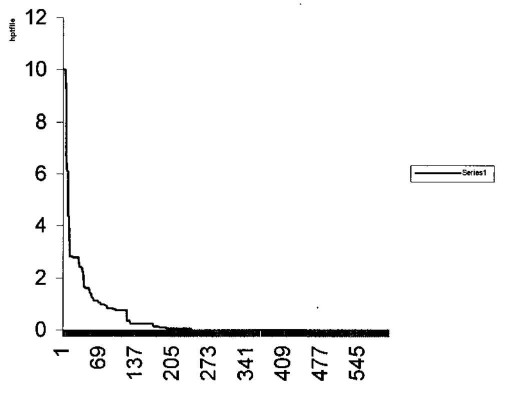
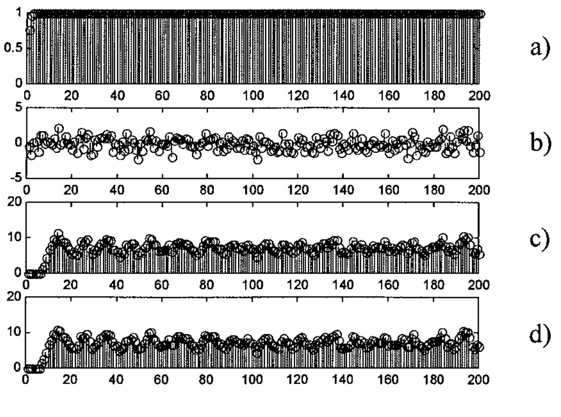


Figure 3: a) error performance curve with the input u[k] = $(1.1)^{-k}$ b) error performance curve with the input u[k] = tanh(k)



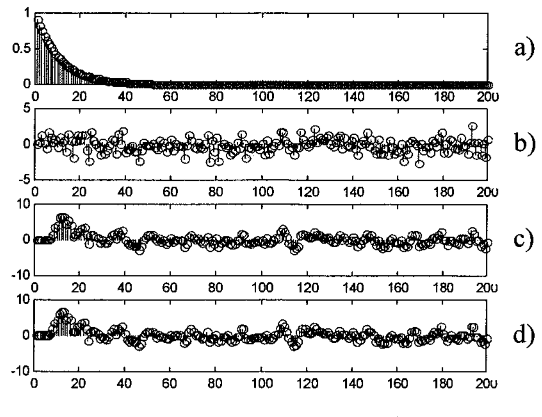


Figure 4: plot of the target system response and its estimated model response by GA.
a) the input to the system u[k] = tanh[k],
b) the Gaussian white noise
c) target system response, d)Estimated system response

Figure 5: plot of the target system response and its estimated model response by GA.
a) the input to the system u[k] = $(1.1)^{-k}$,
b) the Gaussian white noise
c) target system response, d) estimated system response

For comparison with the GA identification technique, the recursive least mean squares method was applied to the same set of data and the best LMS parameter set was found to be [1.502 –0.699 1.000 0.5001 1.001 –1.000 0.200], and the associated error was 7.02e-5. The mean square error of the best-estimated model from the LMS method is significantly less than the error obtained by GA. This result of the LMS method outperforming the GA method is not surprising because from the calculus point of view LMS is the optimum solution for estimating linear systems when the LMS performance measure is used. However, because of numerical problems, the convergence process of the LMS was not stable. In other words, the convergence sequence was not monotonic even for a small number of parameters. As the number of parameters of a model increases, the computational complexity of the LMS method increases rapidly due to matrix multiplication and inversion, while the complexity of the GA is affected very little, since the extra parameters affect only the encoded string which involves very little in the computational process of the algorithm. This is the most important advantage of GA over LMS. Another advantage of GA is that it is much easier to adapt the algorithm to a non-quadratic optimal criterion such as |e|, e^4, etc.

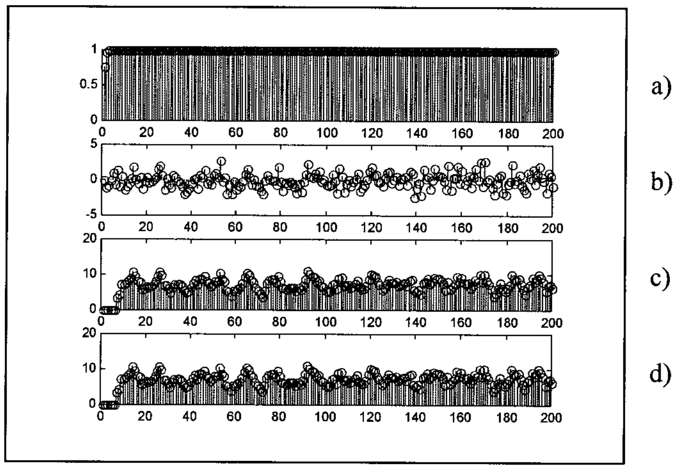
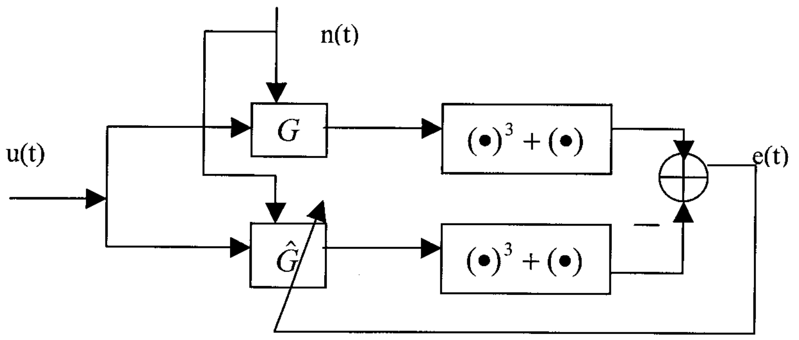


Figure 6: plot of the target system response and its estimated model response by LMS.
a) is the input to the system u[k] = tanh[k], b) is the Gaussian white noise
c) target system response, d)Estimated system response

4. Nonlinear systems

In this section, the GA is applied to identify a non-linear system. The simulation was carried out on a system that is similar to that which was used in the linear example. However, in this case, the output was observed after it had passed through a non-linear block as in the following diagram.



In particular, the following is the nonlinear system model of the system to be identified

$$
\begin{aligned}
x[k] = {} & a_1x[k-1] + a_2x[k-2] + \\
& b_1u[k-5] + b_2u[k-6] + \\
& c_1n[k] - c_2n[k-1] + c_3n[k-2]
\end{aligned} \tag{8}
$$

$$
y[k] = x^3[k] + x[k] \tag{9}
$$

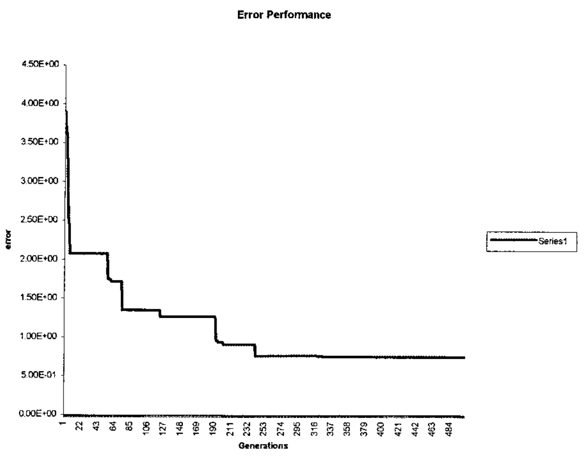
The identification task here is to estimate the parameter set [a_1 a_2 b_1 b_2 c_1 c_2 c_3] optimally given that the observed values $\hat{y}[k]$, and the state values up to k-1 are available. In this simulation, the $x[k]$ are generated by equation (5) and $\hat{y}[k]$ are generated by equation (9). The input $u(k)$ is $(0.9)^k$, and $n(k)$ is again a Gaussian white noise with zero mean and unit variance. Since $y[k]$ is a nonlinear function of $x[k]$, it is not possible to directly use a conventional identification technique such as LMS to estimate the system parameters; therefore, the GA based identification method is applied here. Again we defined the performance measure for the model system as follow:

$$
\mathrm{J} = \sum_{k=1}^{L} (\hat{y}[k] - x^3[k] - x[k])^2 \tag{10}
$$

Using equation (8), J can be rewritten as,

$$
\begin{aligned}
\mathrm{J} = \sum_{k=1}^{L} \{ & (\hat{y}[k] - (a_1x[k-1] + a_2x[k-2] + b_1u[k-5] + b_2u[k-6] \\
& + c_1n[k] + c_2n[k-1] + c_3n[k-2])^3 \\
& - (a_1x[k-1] + a_2x[k-2] + b_1u[k-5] + b_2u[k-6] \\
& + c_1n[k] + c_2n[k-1] + c_3n[k-2])\}
\end{aligned} \tag{11}
$$

As in the linear case, an initial population of 500 models was generated. Each parameter in the model was uniformly distributed in the range of [-2 2]. An offspring was created with the mutation rate of 0.3 and the crossover rate was 0.8. The performance measure was again used as the fitness score for each model. The selection process is stricter as compared to the linear case. Here, only the top 7% of the population was advanced to the next generation, and the rest of the population in the next generation were newly created through the genetic operators. The optimization process evolved for 600 generations. Figure 7 shows the convergence trend of the best model in each generation. The curve converges relatively slowly compared to the linear case. The best model that the GA found after evolving in 600 generations was [1.3332 –0.6097 1.8099 0.3028 0.7970 –0.5731 –0.0528] with the associated error of 7.51e-1. Figure 8 shows the plots of input and output of the target system as well as for the estimated system.



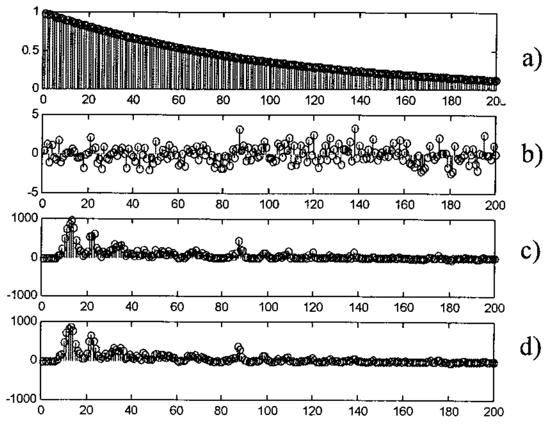


Figure 7: plot of errors during the evolutionary process for a nonlinear system case.

Figure 8: plot of the target system response and its estimated model response by GA.
a) the input to the system u[k] = $(0.99)^k$,
b) the Gaussian white noise
c) target system response, d) estimated system response

For comparison, a calculus-based method was used to identify the model system using the same data set. Consider the system performance measure J, equation (11), as a multivariate function of [a_1 a_2 b_1 b_2 c_1 c_2 c_3], the minimum value of J can be found by solving a system of first order derivative equations for J.

$$\begin{aligned} &\frac{\partial J}{a_1} = 0 \\ &\cdots \\ &\frac{\partial J}{c_3} = 0 \end{aligned} \qquad (12)$$

Solving the system of nonlinear equations (12) by Newton's method, we obtained the solution for the model as [1.5 -0.7 0.747 0.75 1 –1 .2], and the associated error 4.23 e-2. The error obtained by calculus-based is relatively better than the error obtained by GA but there is a trade off as follow: the convergence of the Newton's method depended solely on the supplied initial solution, which was selected by trial and error. For most of the trials for the initial solution, the Newton method failed to converge. The selection of the initial solution was a time consuming process.

5. Conclusions

In this paper, we have addressed the application of the Genetic Algorithm method to the problem of system identification. As a part of the study, the GA identification method was compared the minimum mean square identification method. Our simulation indicates that for identifying linear systems, the LSM method provides a more accurate solution than the GA method. However, because of the complexity and slow convergence, particularly for large order systems, of the LSM method, the GA method has advantages over the LSM both in being less complex and in having a faster convergence time. For large, nonlinear systems, the GA identification method has computational advantages over the nonlinear mean square method because it does not require derivative information on the performance index, which usually involves heavy computation, and the GA can be used to identify a system using a variety of objective functions not just for quadratic criteria.

The GA was applied to linear and nonlinear systems both of which included white noises as inputs. The results showed that the GA worked very well in both cases. The error performance curves indicate that the method converges rapidly. However, during the simulation, we did find that, the convergence of the process is highly sensitive to the GA parameters such at population size, mutation and crossover probabilities, and also to the percentages of the selection process. To overcome this sensitivity, the authors are working to include some of those factors as design parameters. In other word, the GA parameters will be encoded in the individual string and be optimized by the evolutionary process as well. The authors are now studying this change.

Acknowledgements
This work described in this paper was performed at the Center for Integrated Space Microsystems, Jet Propulsion Laboratory, California Institute of Technology and was jointly sponsored by the National Aeronautics and Space Administration and the Defense Advanced Research Projects Agency.

6. References:

[1]. J.-S. R. Jang, C.-T. Sun, E. Mizutani. "Neuro-Fuzzy and Soft Computing, A computational Approach to Learning and Machine Intelligence". Matlab Curriculum Series. Prentice Hall. 1997.

[2]. William S. Levine. "The Control Handbook". CRC Press. IEEE Press. 1995

[3]. Kristinsson, K. and Dumont, G.A. "System identification and control using genetic algorithms". IEEE Trans. Syst., Man and Cyber., 1992, 22(5), 1033-1046

[4]. Fogel, B. D. "Evolutionary Computation: Principles and Practice for Signal Processing". SPIE Press. 2000

[5]. Goldberg, D. "Genetic algorithm in searching, optimization and machine learning". Addison-Wesley, 1989

[6]. Soderstrom, T. and Stoica, P. System Identification, 1989 Prentice Hall .